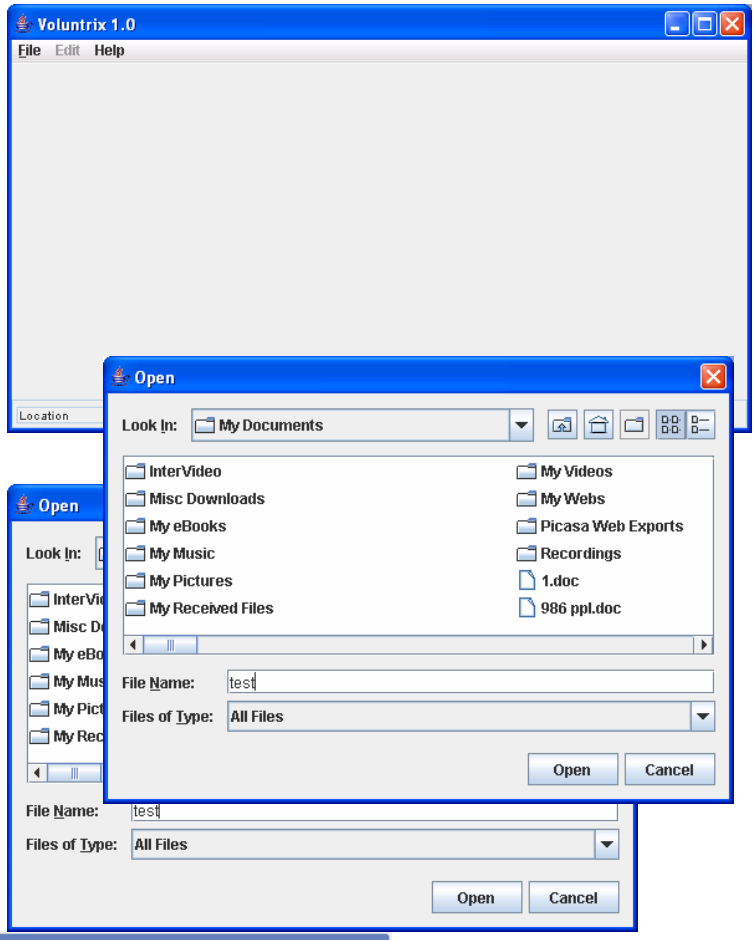
SECTION D

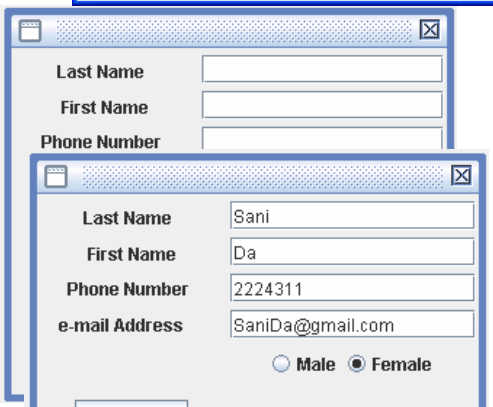
D1. Annotated Hard Copy of the Test Output.

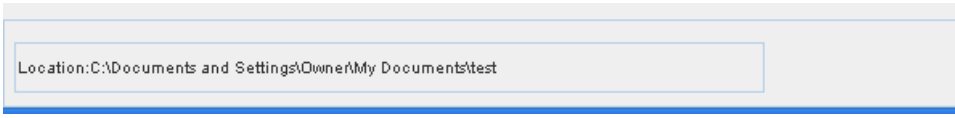
Testing of the Functional Requirements (As described in Section A2)

Testing the Creation of a new File. (A2:1.2.4,2.2,3.1.x)



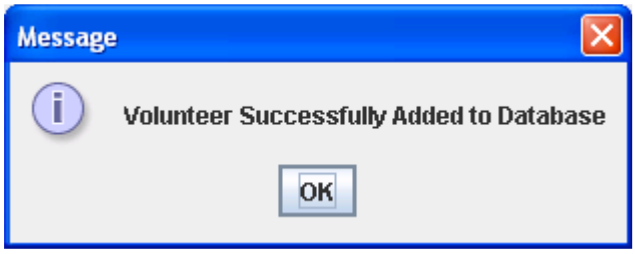
When the program opens up, the user is to click on File>> New/Open File. When this occurs the user can save/open a previously existing file or new file. This creates a new file that is saved on the database, whose location is shown on the location toolbar, and the succesful creation of the file is expressed through a dialog.





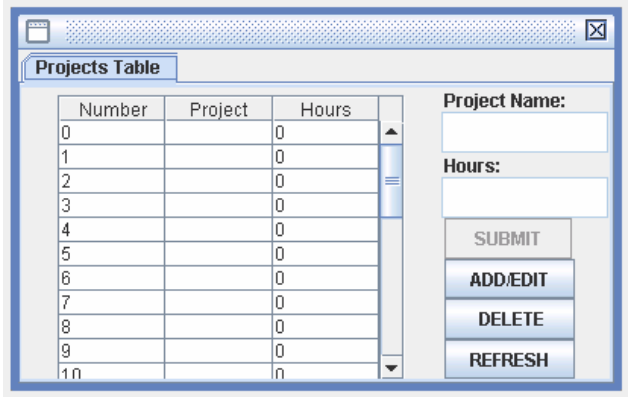
Testing of the Storage of Personal Information for the Volunteer. (A2;3.2.x)

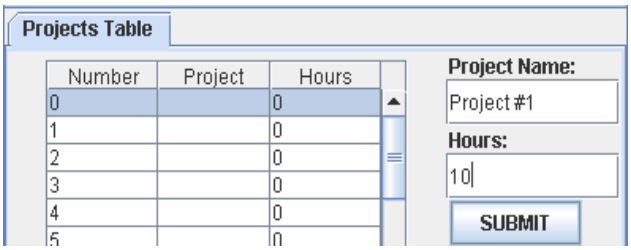
When the edit dialog appears, it allows for the user to input the necessary personal information for a given volunteer. If the data is in the appropriate format for each field, then the user can click the “SUBMIT” button to store this personal information on the file the user had initially selected on the system.

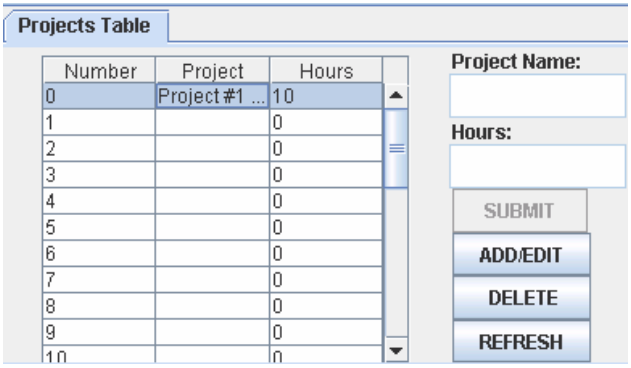


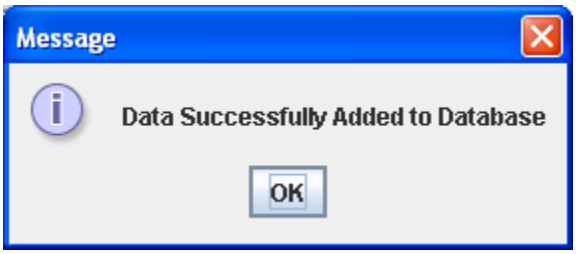
Testing of the Addition of Projects / Hours for a Volunteer. (A2: 3.3,x, 3.4.x)

The user has the ability, to select a given volunteer, and add projects and hours to his/her account. As visible, the user selects a row he/she wants to edit, clicks on the ADD/EDIT button and inputs the data he/she wants to place on the file. As visible, this feature works properly, in that the added hours for the volunteer are properly saved on file.

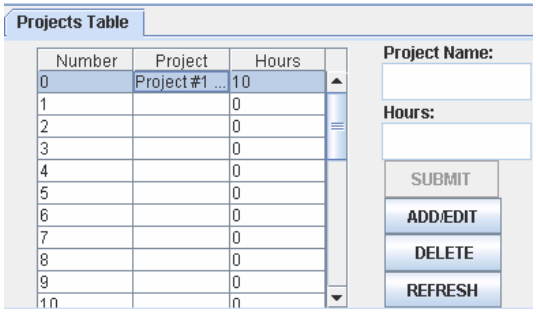




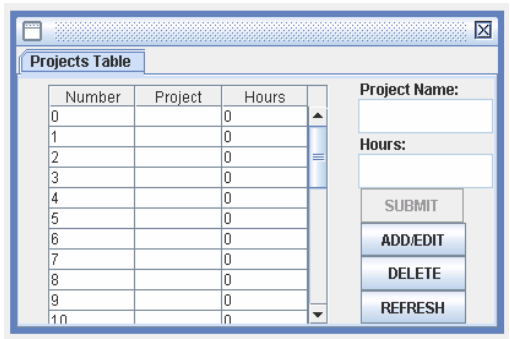




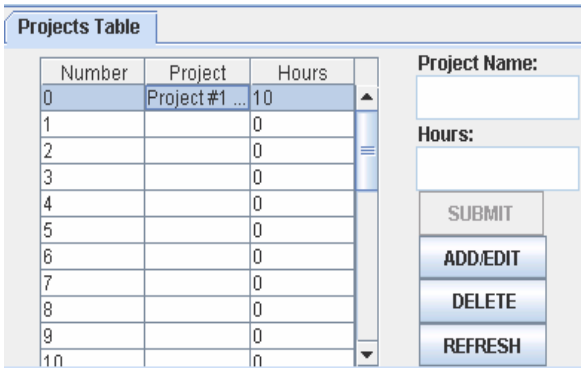
Testing of the Deletion of Projects / Hours for a Volunteer. (A2: 3.3,x, 3.4.x)



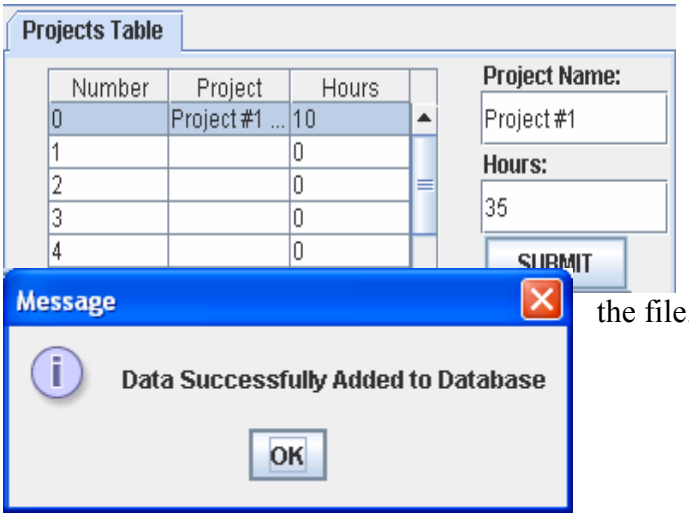
The user has the ability, to select a given volunteer, and delete projects and hours from his/her account. As visible, the user selects a row he/she wants to delete, clicks on the DELETE button. As visible, this feature works properly, in that the row that is to be deleted, is cleared on the file.

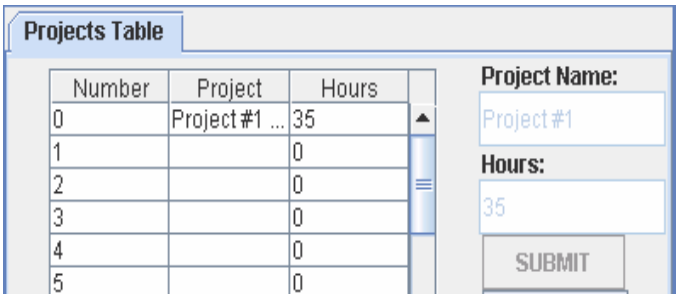


Testing of the Modification of Projects / Hours for a Volunteer. (A2: 3.3,x, 3.4.x)

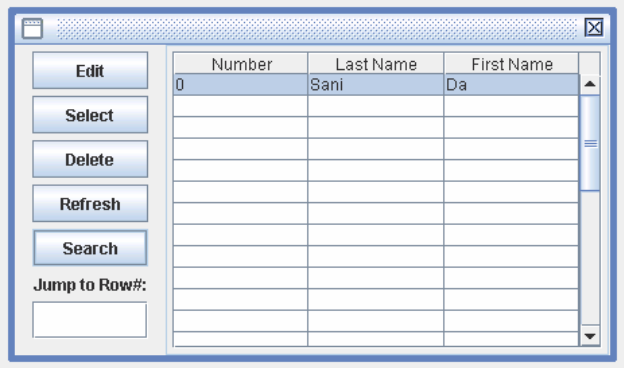


The user has the ability, to select a given volunteer, and edit projects/hours in his/her account. As visible, the user selects a row he/she wants to delete, clicks on the ADD/EDIT button, and fills in the data he/she wants to write to the file. As visible, this feature works properly, in that the Volunteer's data is properly modified, and saved on the file.

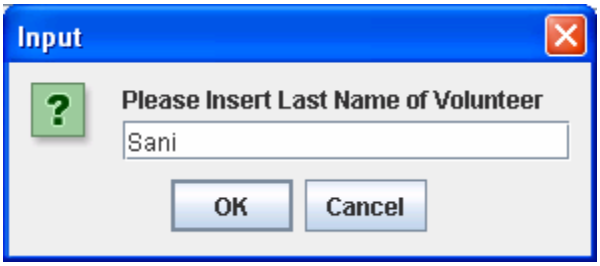


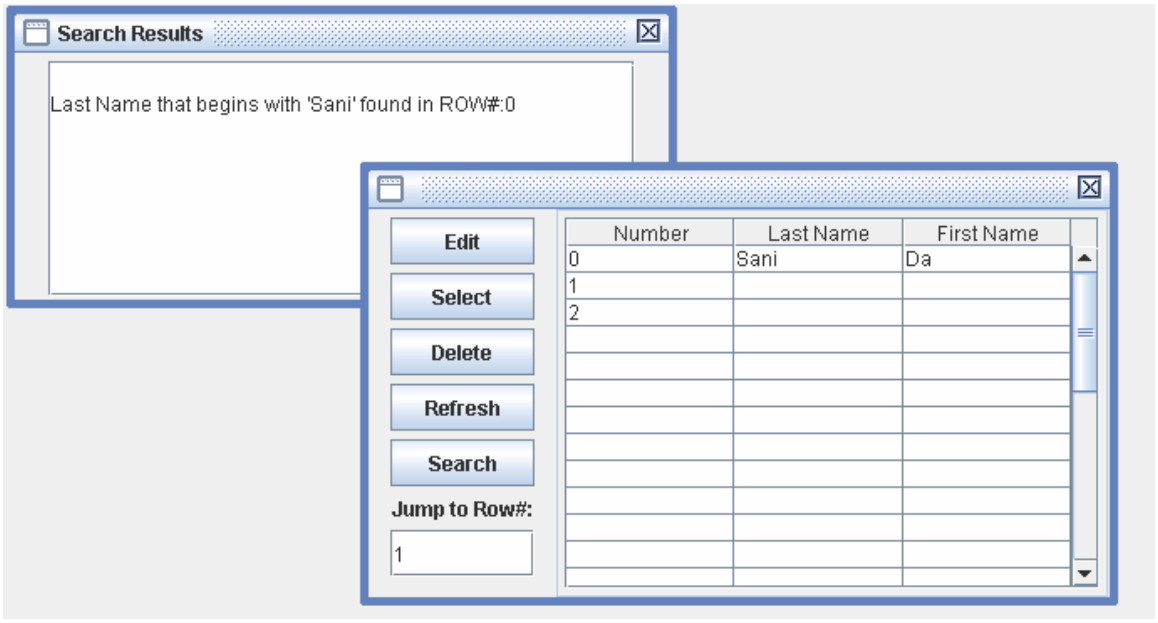


Testing of the Searching Feature. (A2: 3.5,x)



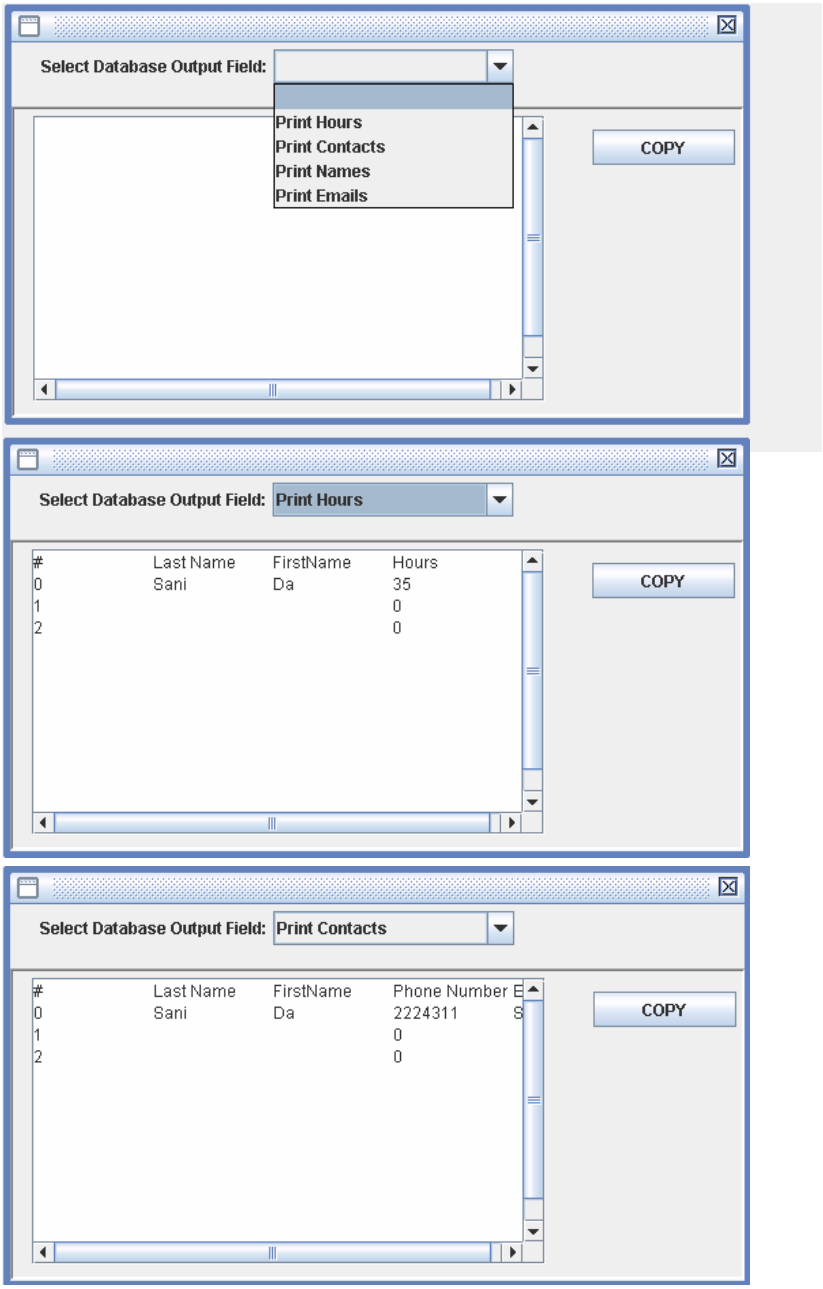
The user is given the ability to search through the Volunteer Database file to find a particular volunteer, based on his/her last name. The user would click on the Search button and input the volunteer's last name. With this done, a dialog appears that gives the location of the volunteer on the table. With the location given, the user can go find the volunteer using the Jump to Row# feature.

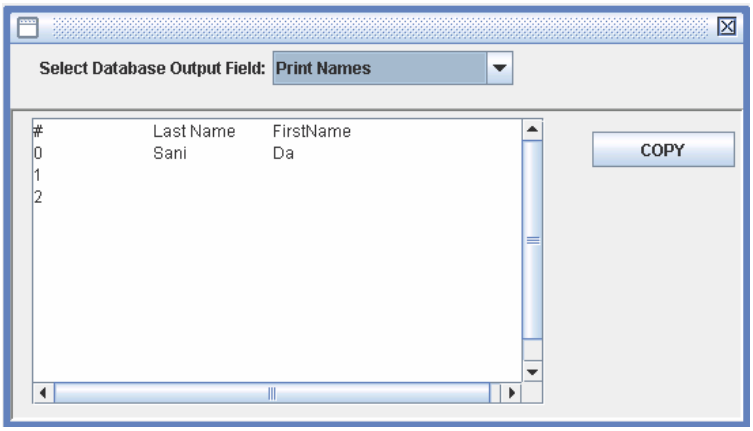


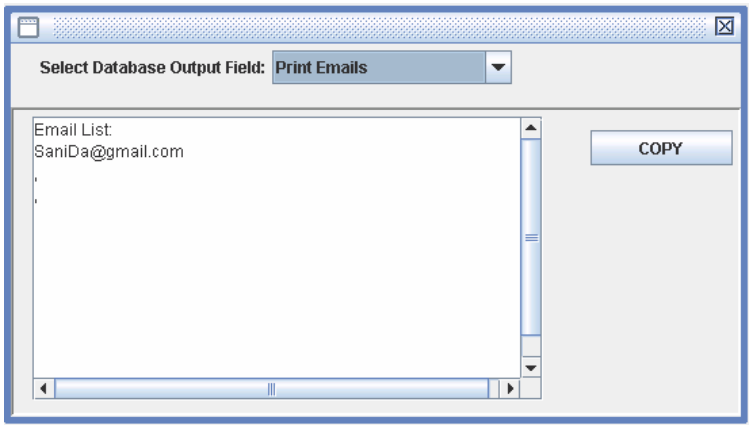


Testing of the Printing Feature. (A2: 3.6.x)

The user is given the ability to print the data within a file, based on all requirements stated in A2:3.6.x.







Testing of the System Context Objectives (As described in Section A2:1.2.1,2.1,2.3)

*Section A2:2.2, is described when testing the open/new file feature

As visible in the screen dumps, the *Voluntrix* is entirely GUI based, and all the functionalities are accessible via menu systems, buttons, and internal frames.

In order to test the time taken to open a file, or create a new file. A console statement was placed when the user selected a file, and one after the file was retrieved to find the

duration of time necessary to retrieve a file on a modern computer (Pentium IV with 2.40 gHz and a 512 mB RAM running Java version 1.5.0).

Code Excerpt
Location: *MainApplication.java*
Comments: *Notice how the system clock is used to output the time, after the file is retrieved*

```
try{
        System.out.println("Current Time in milli seconds:" +
                System.currentTimeMillis());
        file = new RandomAccessFile(fileName,"rw");
        MsgField.setText("Location:"+fileName.toString());
        file.writeLong(pointer);
        System.out.println("Current Time in milli
                seconds:"+System.currentTimeMillis());

                }
```

Console Output:

```
Current Time in milli seconds:1143653382906
Current Time in milli seconds:1143653382906
```

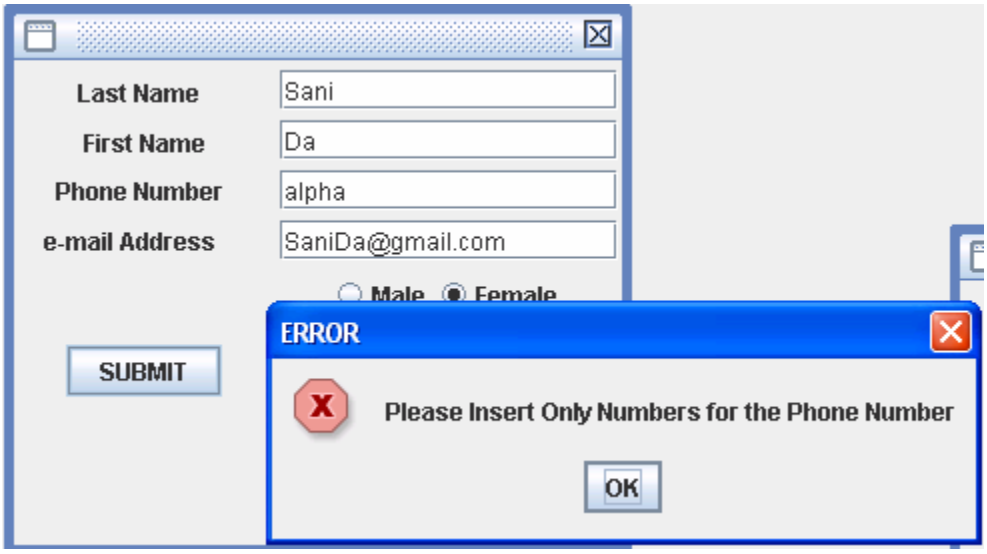
As the console output suggests, the time taken to access the random access file is very small; thereby, successfully completing objective A2: 2.3.

Testing of the Quality Requirements (As described in Section A2:4.x,6.3)

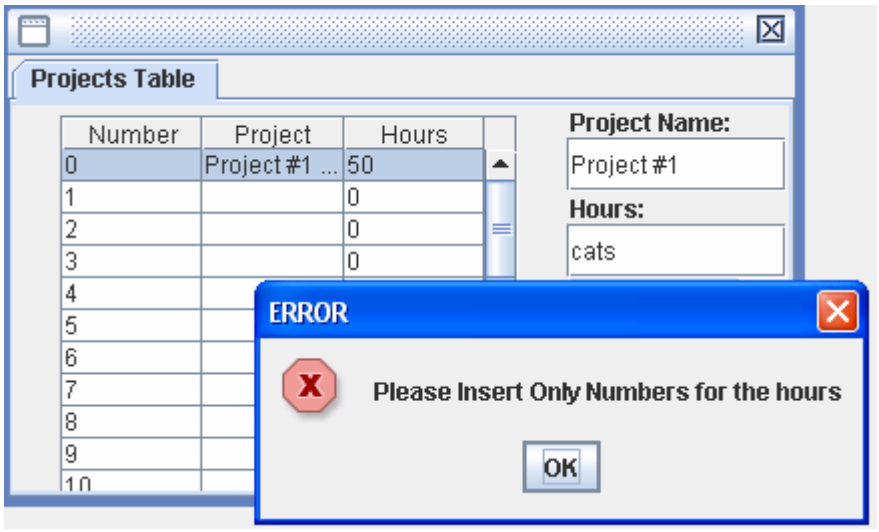
The reliability objectives as described in A2:4.1 are thoroughly detailed in section C3. However, two major categories of errors shall be discussed here:

1. Invalid Data Handling

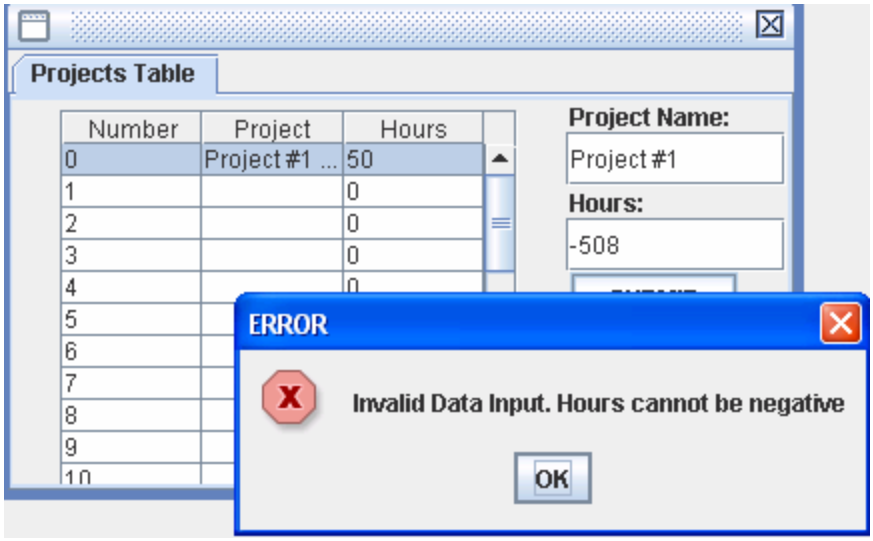
    a. An error message dialog successfully appears when a user inputs a non-numeric entity in the phone number dialog and submits it.



    b. An error message dialog successfully appears when a user inputs a non-numeric entity for the hours of a volunteer and submits it.

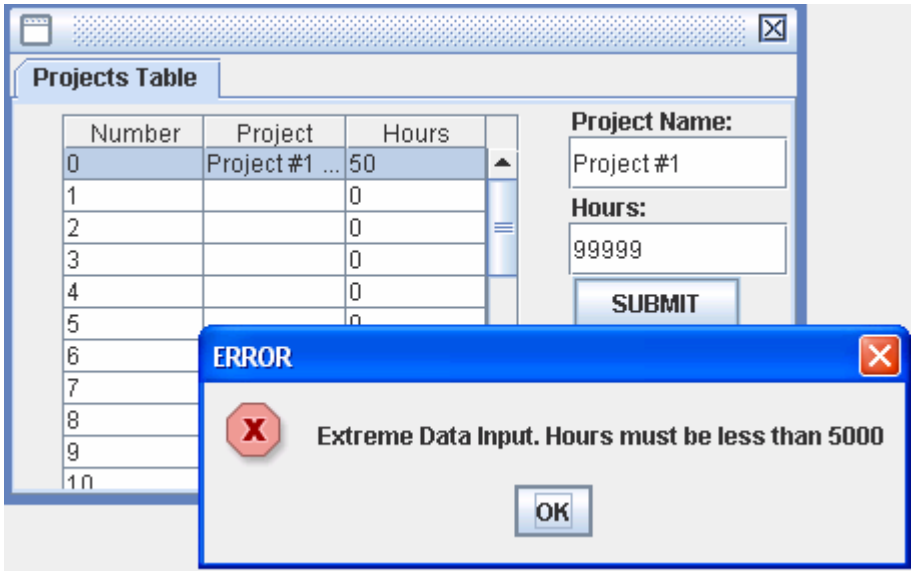


c. An error message dialog successfully appears when a user inputs a negative numeric value while inputting Volunteer hours.



2. Extreme Data Handling

a. If the user inputs more than 5,000 hours for a particular project an error message successfully appears to indicate an extreme data input.



In terms of portability, since this program is programmed in JAVA it is assumed that provided that a JVM is present on a machine, the *Voluntrix* can run on any platform (Linux, Mac, etc.). (A2:6.2)

In terms of modifiability (as detailed in section A2:4.3.x) the following is present:

- Easy to understand source code, along with the presence of detailed UML Class Diagrams and Modular Design Diagrams (4.3.1)
- The presence of very detailed JavaDocs (4.3.2) and a User's Manual
- The presence of comments (both Javadoc and simple comments (//…)) throughout the source code (4.3.3).

Testing of the System Objectives (As described in Section A2:1.x)

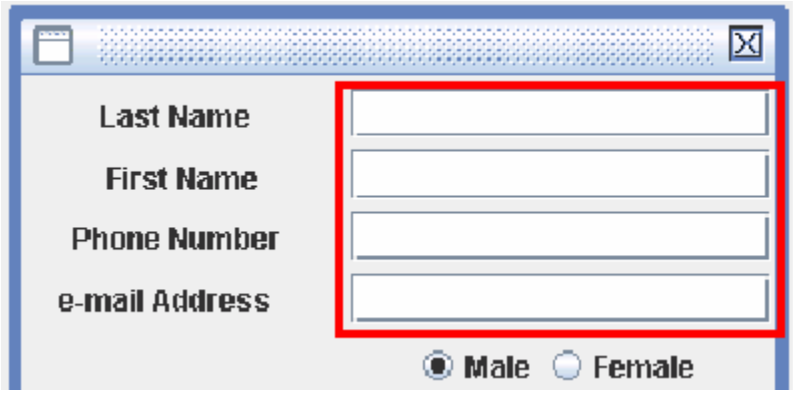
As visible through the previous testing scenarios, the *Voluntrix* is successful in maintaining and handling information for an array of volunteers for a particular organization (section 1.2).

In terms of Usability, all of sections A2:1.2 are detailed in section C2. However there is a need to elaborate on section 2.1.1: *Ease of Input.*

Ease of Input: (As described in Section A2:2.1.1.x)

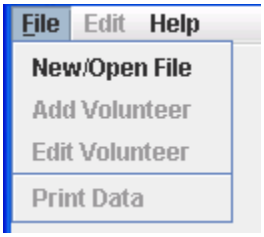
The key input features detailed in section A2:2.1.1 are tested below:

1. TextBoxes:



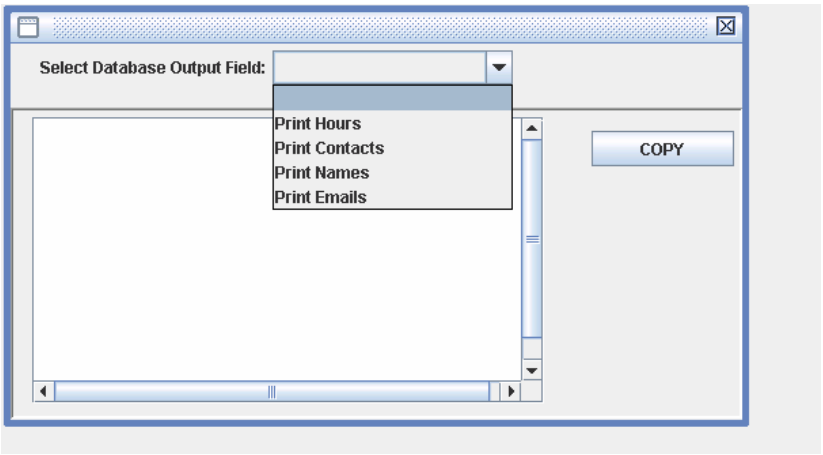
Textboxes throughout the program successfully allow the user to directly input data for a particular field.

2. Dropdown Menus



These components successfully handle user input, in that by selecting a menu item the user invokes a particular function.

3. Dropdown Lists



This component allows the user to select one of several list elements, which would serve as an action to invoke particular print features.

4. RadioButtons

This component successfully allows the user to select only one or the other option when inputting a volunteer's sex.

Summary of Testing

As detailed in this section, all the key objectives put in place in Section A2 have been completed thoroughly. Due to the lack of time, the objectives in the Future Requirements Section of A2 (A2.5.x) have not been completed. However, due to the modifiability of the source code these functions can be added in the near future.

D2. Evaluation of Solutions.

Effectiveness

The Voluntrix accomplished all the goals as set out and established in the A2 list of objectives. This would deem the solution as effective based on its relations to the original description of the project. The application that resulted from this dossier is simple, versatile, robust, and effective in managing volunteer information for volunteering organizations. A simple GUI system makes it easy for a person with minimal computer skills to work with the program and successfully manage a volunteer organization's volunteers and activities. In addition to management of the database, the user can scan through the database quite easily to find a particular volunteer, and he/she can print out information from the file and copy it to the clipboard to move it where they choose to.

The simplicity of the program, along with a solid user's manual and error handling capabilities almost make it impossible for the program to crash .

Efficiency

Efficiency can be considered the production of desired results with the implementation of minimal waste of time, effort, or skill[1]. In terms of programmer efficiency, factors like the ease of modifiability, robustness of the program, exception handling tools, and effectiveness of algorithms comes into consideration. Machine

[1] Efficiency. (n.d.). *Roget's II: The New Thesaurus, Third Edition.* Retrieved March 29, 2006, from Answers.com Web site: http://www.answers.com/topic/efficiency

efficiency on the other hand refers to how the computer performs tasks and depends on factors like the RAM, operating system, CPU, and disk space.

In terms of recursion in this program while searching, since the volunteers are not sorted alphabetically or numerically (first into the file, is the first on the file) a counter has to systematically go through the file to find a particular volunteer. Although this is great for a small volunteer organization with a limited number of volunteer (as was the end user for the *Voluntrix)*, if this program were to be marketed to a large volunteering organization with thousands of volunteers, searching systematically from one record to the other till the end of the file could be very time consuming. This system of systematically scanning the file for volunteers is also used in the printing feature, where every record is red and placed on a JTextArea in a – somewhat – tabular format.

Big O notation, is a theoretical measure to give an idea of the execution of an algorithm in terms of the time or memory required, given a problem of size n. In terms of the efficiency of the searching and the printing algorithms for the *Voluntrix*, there is no constant time for each search, since once the search finds a volunteer it stops; however, the maximum time it can take for a given file is always constant (since it looks through each record). However, if more intricate algorithms are used, the efficiency of the searching process can be made better and less time consuming. However, this would not make a tremendous difference in efficiency for a relatively small volunteer organization using this program.

In terms of garbage collection – or the freeing of objects that are no longer referenced by the program. Garbage collection in java is automatic, since the JVM's heap maintains and stores all the objects initialized and created by a Java application that is executing. Because JAVA was used the concept of garbage collection was not of prime importance; however, in a language like C this concept would have to be analyzed thoroughly.

Alternate Processes and Improvements to the Solution and Design Process

In terms of the design process, if the *Voluntrix* had to be created from scratch it is high recommended that the entire GUI of the program be sketched, and that this design matches all the UML diagrams created. If this were to be followed the program's layout is known, so there is no need to worry about whether the whole thing would work at the end. Another improvement to the design process would be doing the Javadocs before actually programming, in order to add a certain degree of structure during the programming phase. This improvement is effective in that it is a reminder to the programmer as to how to program each module/algorithm.

In terms of improving the efficiency of the solution they key place to improve would be in regard to the searching algorithm. Currently the systematic scanning of the file seems to work; however, in larger files with more volunteers it would be tremendously difficult to find a certain record. To address this, an improvement that can be made is sorting all the volunteers on the file, and then creating an algorithm that can easily move through the database since the records are already sorted. In addition to this,

further improvements that can be made to the *Voluntrix* include the ability to search the database using any key (not just the last name).

In addition to the aforementioned improvements to the *Voluntrix* further improvements are listen in A2: 5.x.

Alternate Approaches to Solving the Problem

Rather than using a direct access system to manage volunteer information, the program could have implemented a linked list. An ADT like the linked list would have had several nodes, each one representing a volunteer. In addition to a single linked list, it is also possible to generate a tree structure based where each node represents a volunteer. Using a binary tree, searching capabilities can be made quite efficient. Although these are alternate approaches to solving the problem, for the problem that was initially given a direct access system of file handling is great.

D2. User's Manual

VOLUNTRIX

USER DOCUMENTATION

TABLE OF CONTENTS

D.3.1:SYSTEM REQUIREMENTS **20**
D.3.1:SYSTEM REQUIREMENTS **20**
D.3.2:INSTALLATION**:** **20**
D.3.3:OVERVIEW**:**..... **20**
D.3.5:HOW TO STARTUP:..... **21**
D.3.6:USING HELP:..... **22**
D.3.7:HOW TO CREATE A NEW FILE: **22**
D.3.8:HOW TO CREATE/ SAVE A RECORD **23**
D.3.9:HOW TO RETRIEVE AND EDIT A RECORD **24**
D.3.10:HOW TO DELETE A RECORD **27**
D.3.11:HOW TO ADD PROJECTS AND HOURS TO VOLUNTEERS **28**
D.3.12:HOW TO EDIT PROJECTS AND HOURS TO VOLUNTEERS..... **30**
D.3.13:HOW TO DELETE PROJECTS AND HOURS TO VOLUNTEERS..... **31**
D.3.14:HOW TO PRINT OUT DATA FROM THE DATABASE **32**
D.3.15:HOW TO COPY AND PASTE DATA FROM THE DATABASE..... **33**
D.3.16:CLOSING THE PROGRAM **33**
D.3.17:TECHNICAL INFORMATION **34**
D.3.18:UPDATES **34**
D.3.19:FAQ (FREQUENTLY ASKED QUESTIONS)..... **34**
D.3.20:SUPPORT **34**

D.3.1:System Requirements

- **System** – Any system capable of supporting and executing .jar files (*Program of choice: Java 2 Runtime Environment, Standard Edition (JRE)).*
- **Processor** – Pentium I with 200Mhz or higher (*recommended)*
- **Monitor** – Recommended Resolution of 800x600 or higher
- **Memory** – Recommended 32MB higher
- **Hard disk space** – Recommended 100MB or higher
- **Hardware** – Mouse and keyboard (*it is assumed the aforementioned requirements have been met).*

D.3.2:Installation: (on a Windows Platform)

1. Insure you have J2SE 5.0 on your machine, since it is necessary to run the program
    - If you don't download it from the Sun Microsystems website: java.sun.com.
2. Insert the CD-ROM with the program into the CD-ROM drive.
3. Click on Start then navigate to My Computer
4. When the My Computer Folder opens up, double click on the icon representing the CD-ROM drive.
5. When the folder representing the contents of the CD-ROM are open, double click on the file "Voluntrix.jar".
6. Observe section D.3.5 to see how to startup the program.

D.3.3:Overview:

Voluntrix allows for a volunteer organization to manage volunteer information quickly and efficiently, in order to keep track of all the volunteers' activities with the organization.

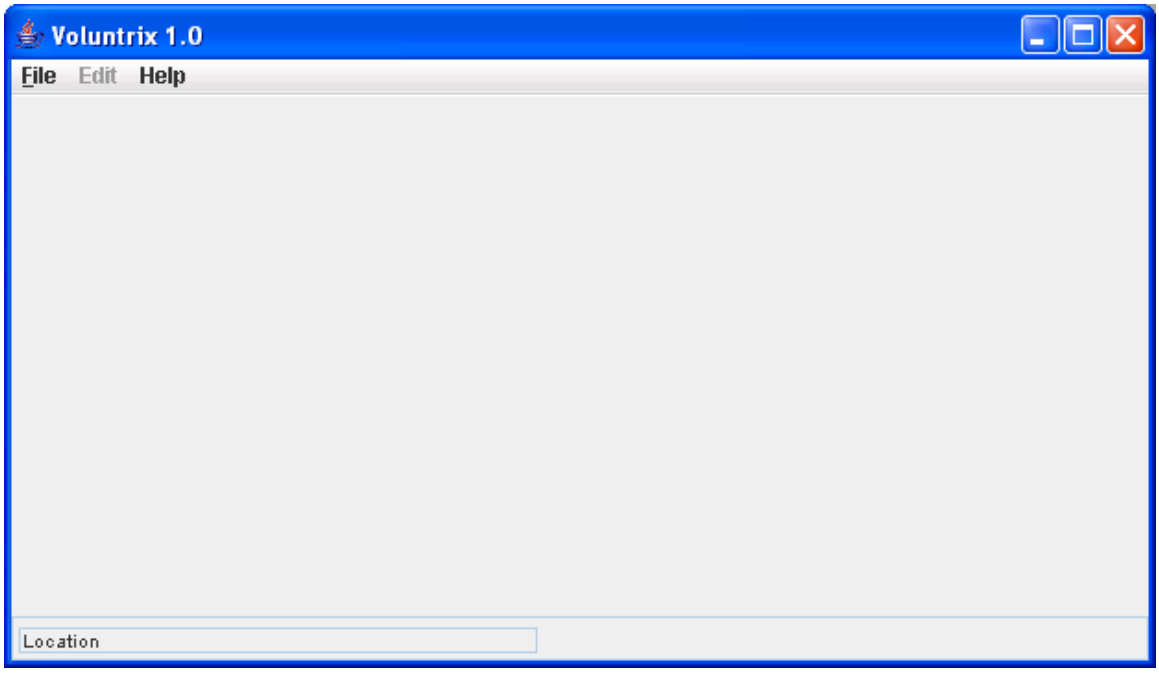
D.3.4:Features / Options:

- Addition, editing, and deletion of volunteers.
- Management of personal information for each volunteer.
- Ability to add, edit, and delete projects and hours for each volunteer.
- Ability to search through the entire list of volunteers, for a given 'Last Name'.
- Ability to print out the data held within a specific data file.
- Ability to copy the printed data outputted by the program to the system clipboard.

D.3.5:How to Startup:

1. Ensure that all the System Requirements for the *Voluntrix* are met.
2. Navigate to the folder where the *Voluntrix* is located, and double click on the executable *.jar file.
    a. **NOTE:** If the *Voluntrix does not open up, run the* command prompt, navigate to the installation folder, and then use the "`jar -jar filename.jar`" instruction method

If everything goes well, the *Voluntrix* should open up and it's main window should be displayed on the monitor.

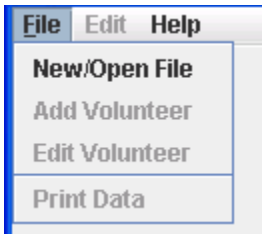


D.3.6:Using Help:

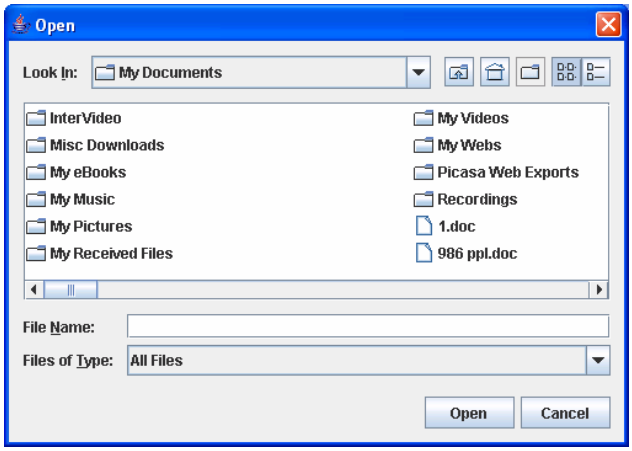
Click on Help in the Menu Bar

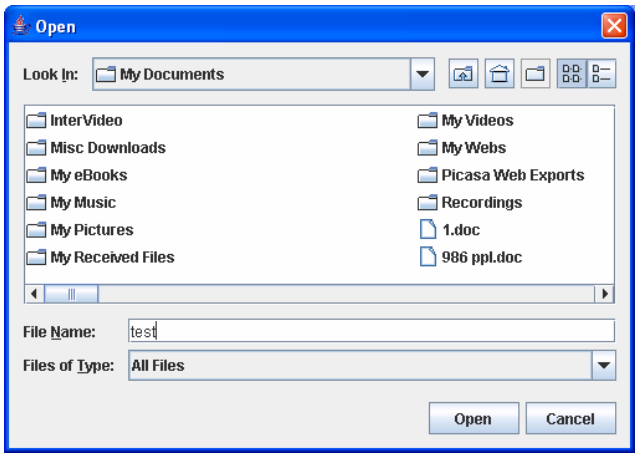
D.3.7:How to Create a New File:

1. Click on **File** (Ctrl+F) and click on **New/Open File**

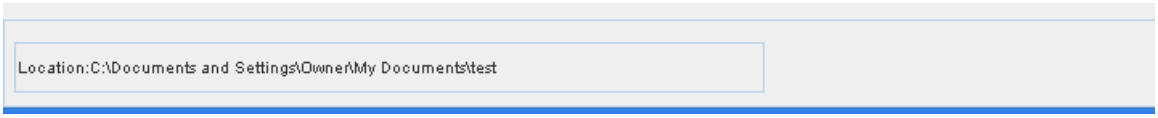


2. Navigate to the folder you would like your data file to be located in, and click "OPEN" in the File Chooser Popup.





3. You will return to the main window, and the path of your file is located at the bottom of the window, in a location bar.

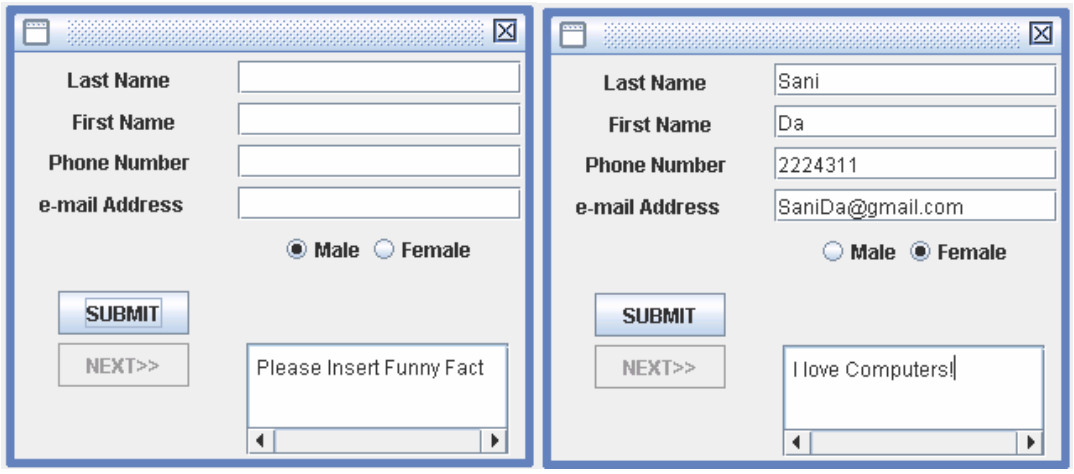


D.3.8:How to Create/ Save a Record

1. Make sure that a new file has been created, or an existing volunteer data file has been opened.
2. Click on File (Ctrl+F) and click on **Add Volunteer**.

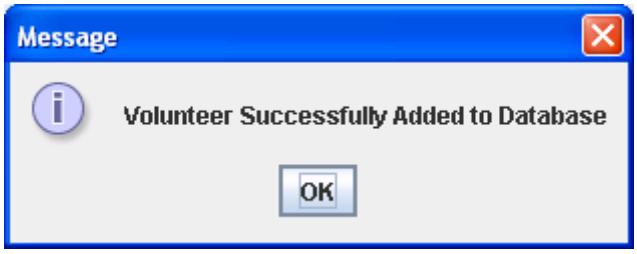
3. In the popup input dialog that shows up, fill out the required information.

   (**NOTE**: the Phone Number input has to be purely numerical)



4. Once the data has been filled in, click the **Submit** button.

   a. This allows for the record to be saved into the file, and if the save is successful a dialog message will appear.

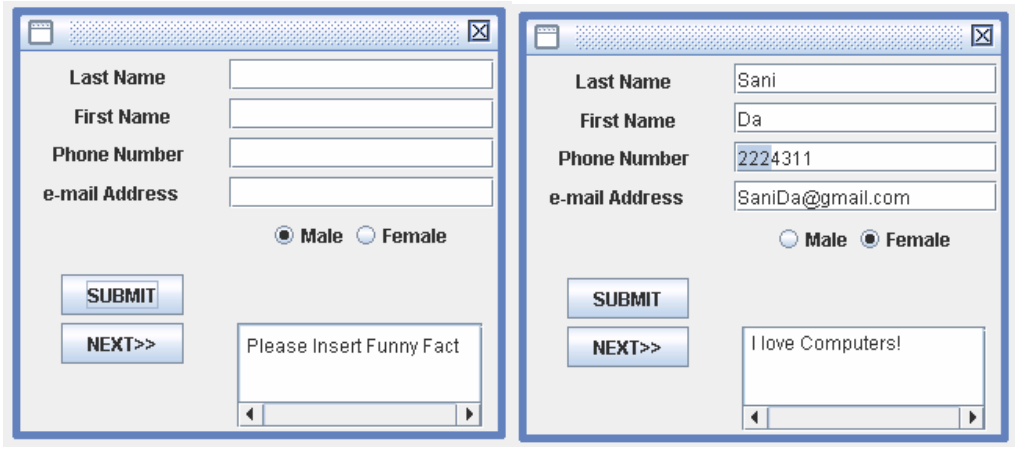


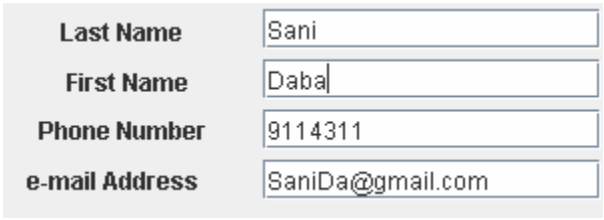
D.3.9:How to Retrieve and Edit a Record

Editing a record can be done one of two ways.

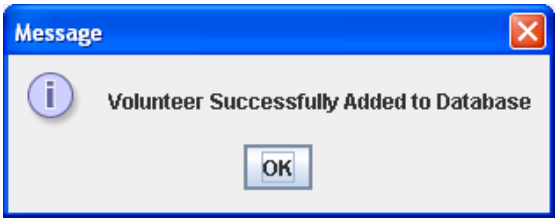
Method I

1. Click on File (Ctrl+F) and click on **Edit Volunteer**.
2. In the popup input dialog that shows up, click on the **NEXT>>** button until the record you were looking for is found.



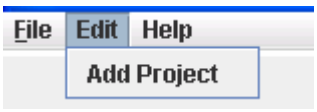


3. Once the record you wanted is reached, you can make the desired change and click on SUBMIT.
    a. This allows for the record to be saved into the file, and if the save is successful a dialog message will appear.

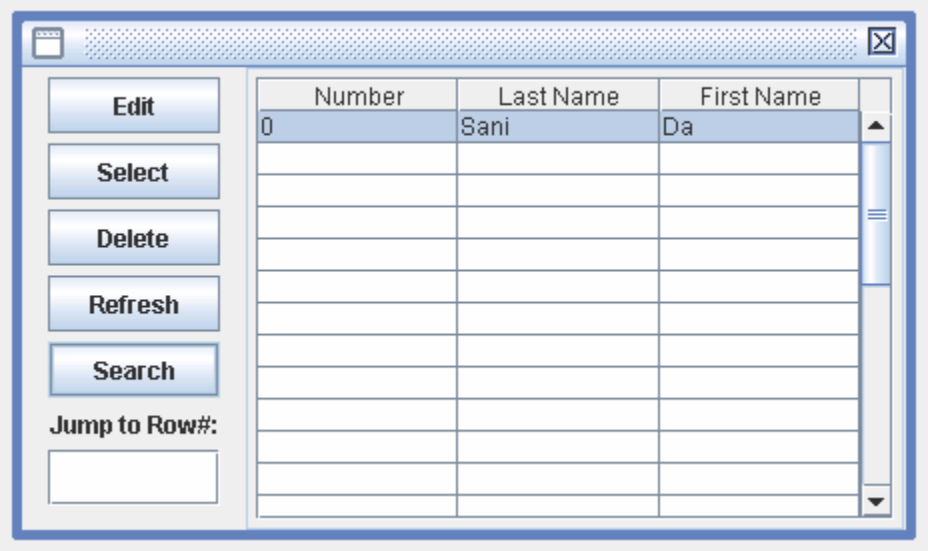


Method II

4. Click on Edit in the menu bar and click on **Add Project**.



5. To retrieve a specific record, one of two things can be done:



   a. In the popup input dialog that shows up, scroll through the Table that is generated to find the desired record, and select the specific row.

   b. **Search**

      i. Click on the **Search** button

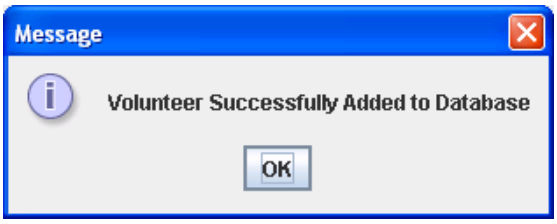
      ii. Insert the name of the Volunteer, in the input dialog that pops up and click **OK.**

      iii. Based on the results printed in the new popup, go the row number/s you were looking for using the **Jump to Row** text field.

6. Once the desired row is selected, click on **Edit**

7. In the popup dialog that shows up, make the desired change/changes and click Submit.

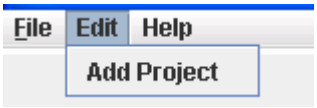
a. This allows for the record to be saved into the file, and if the save is successful a dialog message will appear.



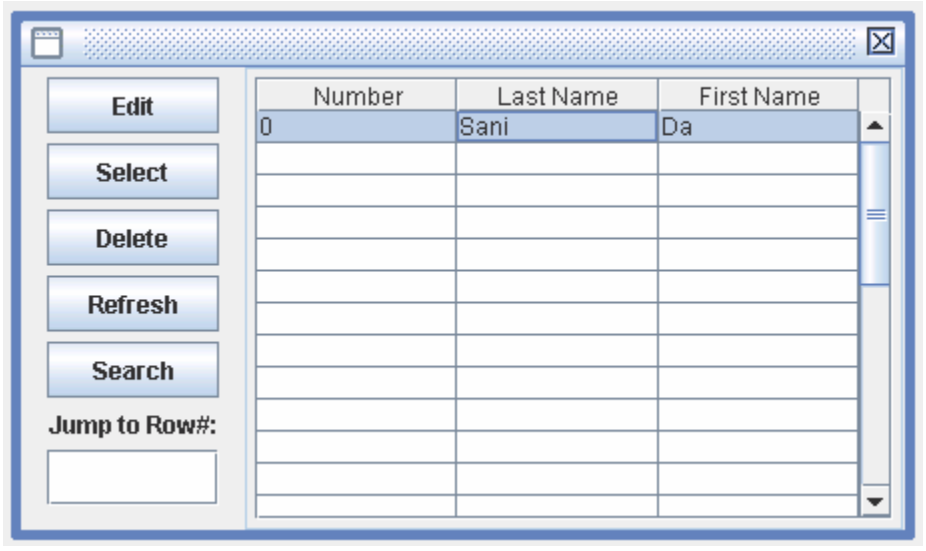
8. Click the **Refresh** button on the Add Project window, for the table to update.

D.3.10:How to Delete a Record

1. Click on Edit in the menu bar and click on **Add Project**.



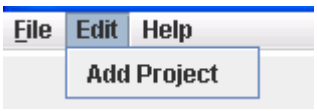
2. Select a specific record (Observe D.3.7)



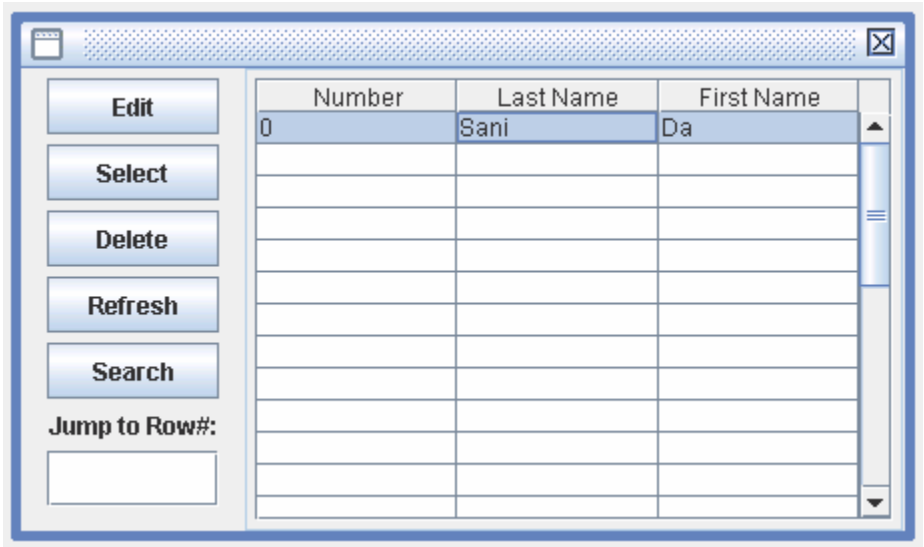
3. Click on the **Delete** button.
    a. This command causes the cell to be cleared of all its information, and the data for that specific volunteer is removed from the file.

D.3.11:How to Add Projects and Hours to Volunteers

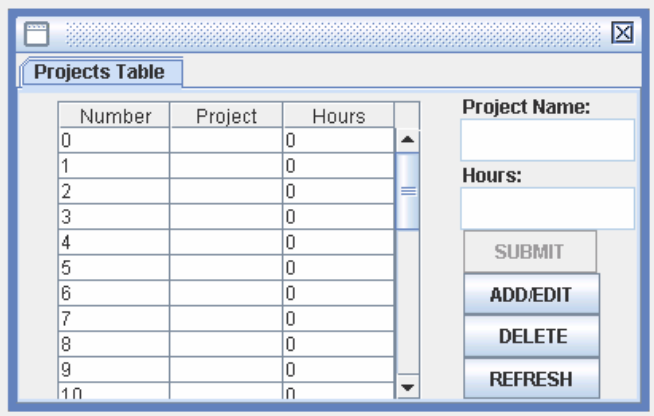
1. Click on Edit in the menu bar and click on **Add Project**.



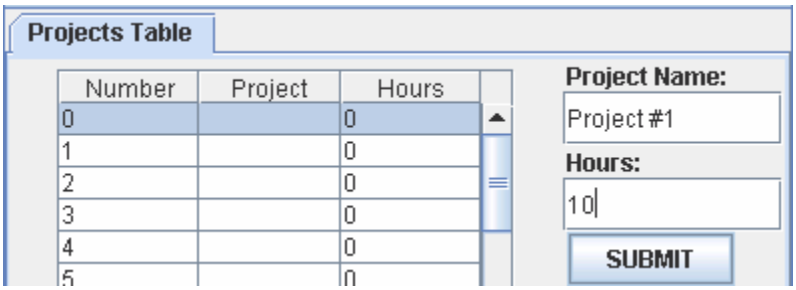
2. Select a specific record (Observe D.3.7)



3. Click on the **Select** button

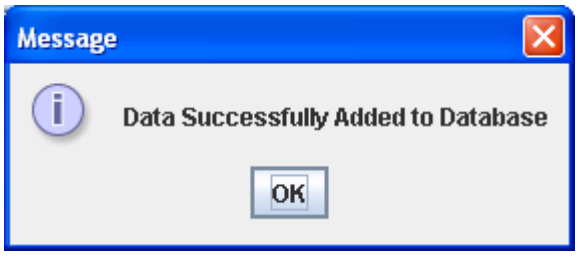


4. On the popup dialog that shows up, select one of the 30 cells on the table, and click the **Add/Edit** button.



5. Type the given Project Name and the total number of Hours the Volunteer has submitted, and Click **SUBMIT.**

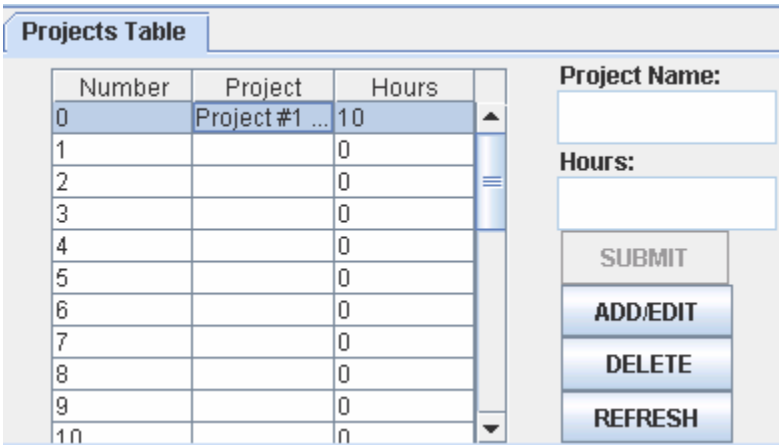
    b. This allows for the data to be saved for a given volunteer into the file, and if the save is successful a dialog message will appear.



6. Click on the **REFRESH** button in the Select Dialog window, for the graph of projects and hours collected to be updated in real time.

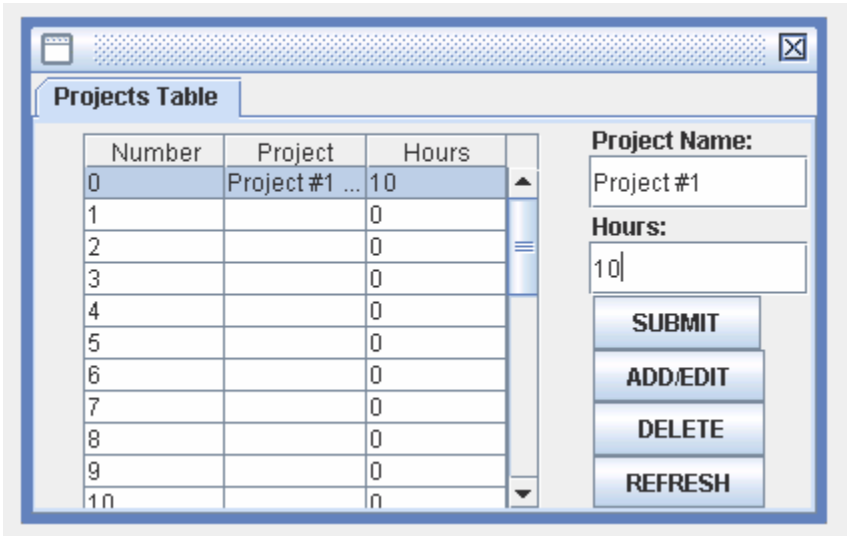
D.3.12:How to Edit Projects and Hours to Volunteers

1. Follow steps 1-3 in section D.3:11.



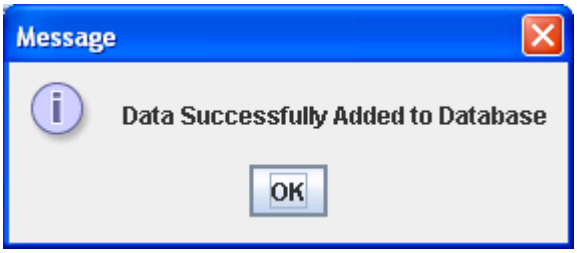
2. Select a cell with previous existing data that requires editing in the table.

3. Click the **Add/Edit** button.



4. Make the required changes to the Project Name or the Hours text boxes. And click the **SUBMIT** button.

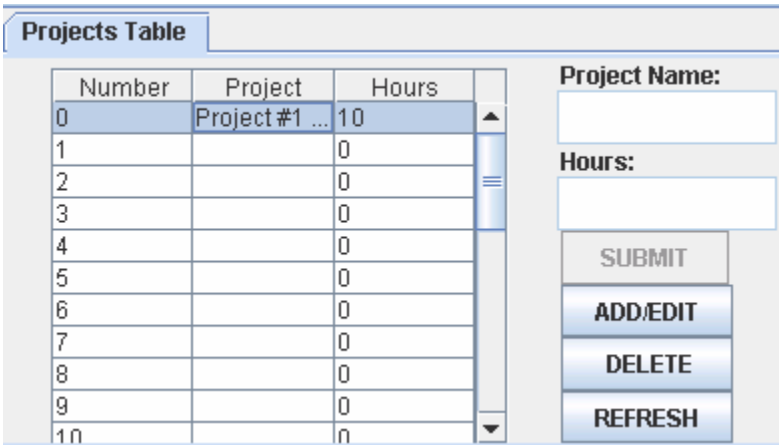
    c. This allows for the data to be saved for a given volunteer into the file, and if the save is successful a dialog message will appear.



5. Click on the **REFRESH** button in the Select Dialog window, for the graph of projects and hours collected to be updated in real time.

D.3.13:How to Delete Projects and Hours to Volunteers

1. Follow steps 1-3 in section D.3:11.

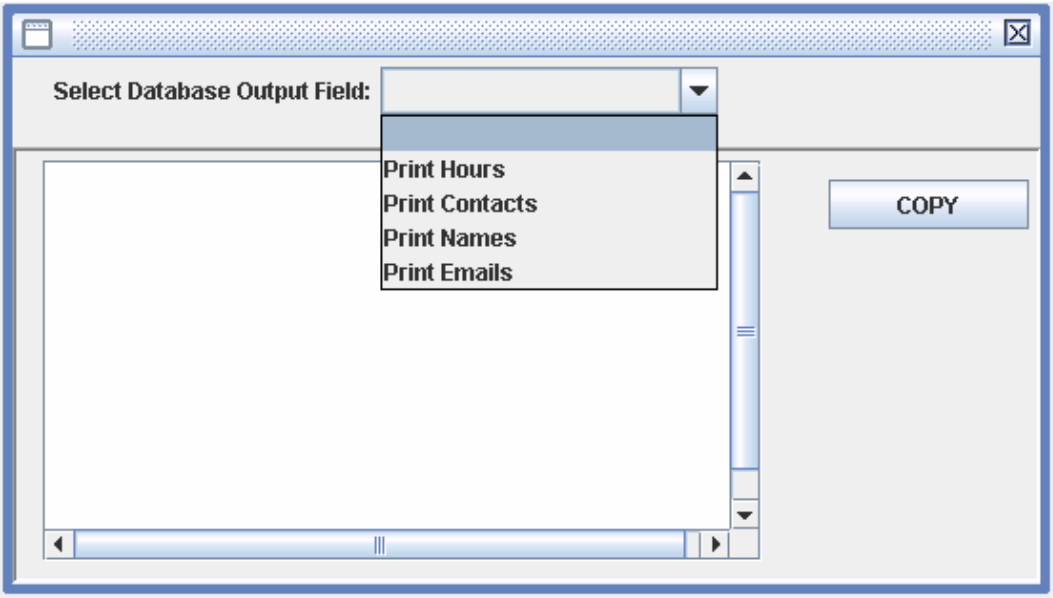


2. Select a cell with previous existing data that needs to be deleted

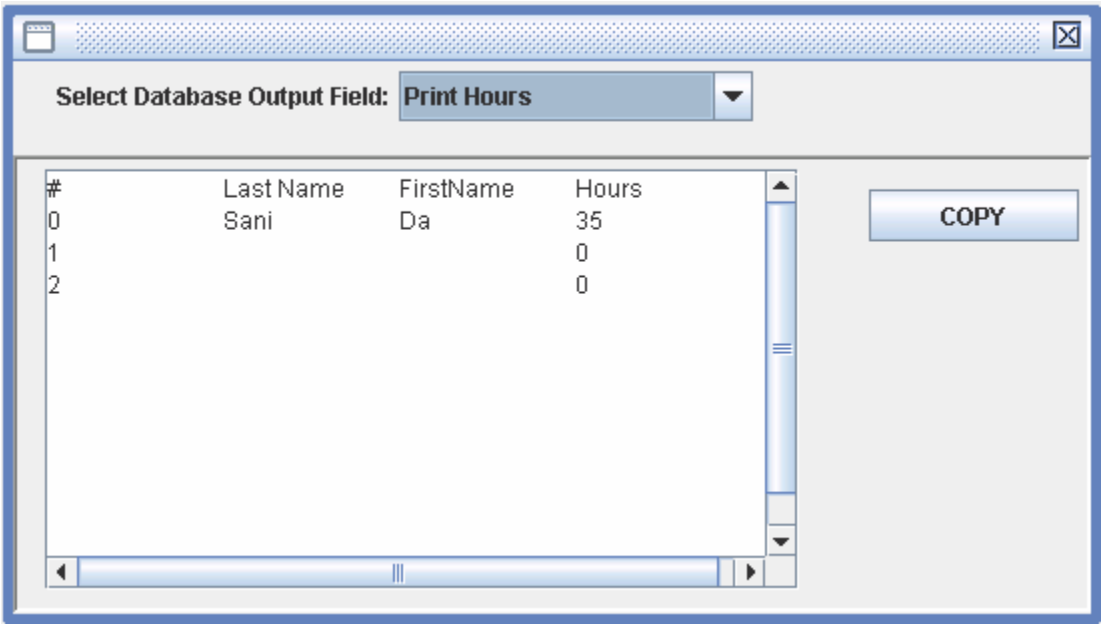
3. Click the **DELETE** button.

D.3.14:How to Print out Data from the Database

1. Click on File in the menu bar and click on **Print Data.**



2. On the Print popup window that emerges, Select the type of data that you would like to print.



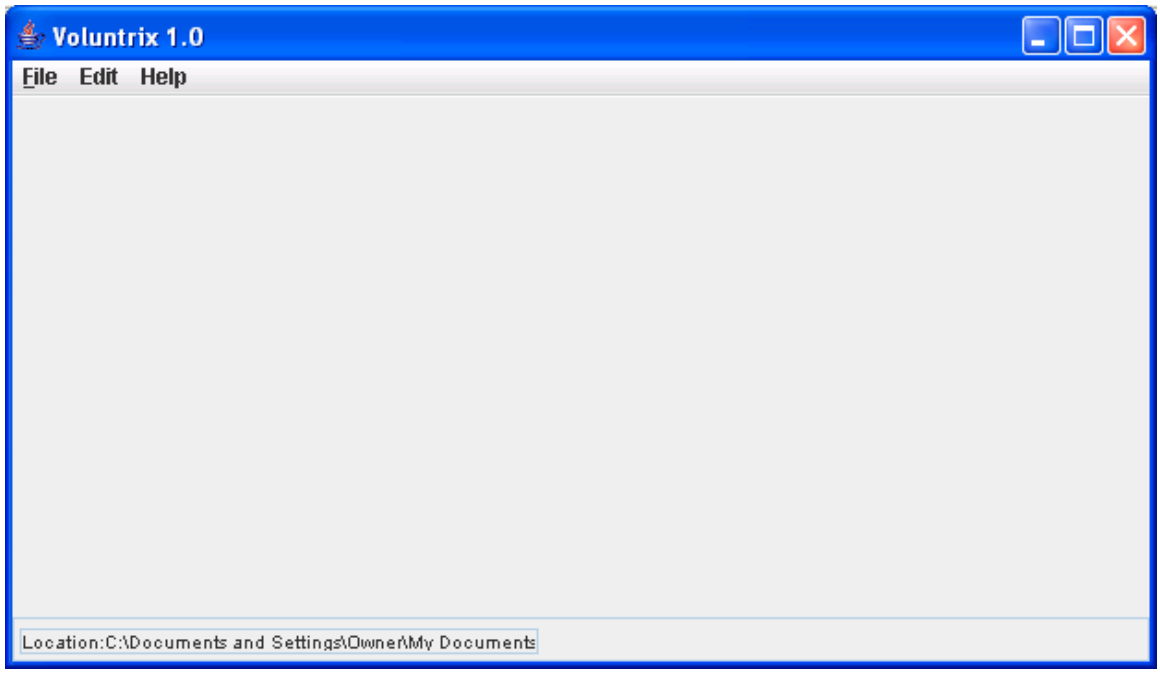
   a. This action causes the data to be outputted automatically.

D.3.15:How to Copy and Paste Data from the Database

1. Follow steps 1,2 in section D.3.12.
2. When the appropriate data is displayed in the window, click the **COPY** button.
3. Go to any text processing program or spreadsheet to paste the data.
    a. In Microsoft Excel 97/2000/XP the data is aligned properly into the given cells to facilitate further actions.

D.3.16:Closing the Program

1. Go to the main application.



2. *Two possible options*:
    a. Click the X on the top right hand side of the window.
    b. Press Alt+F4

D.3.17:Technical Information

Voluntrix is written in the JAVA programming language with Sun's JDK 5.0. It was developed using Eclipse 3.1 on a computer running Windows XP.

D.3.18:Updates

Updated versions of the Voluntrix are available through the developer. Eventually, the Voluntrix should be available from an online source for international availability and use.

D.3.19:FAQ (Frequently Asked Questions)

1. **Is the Voluntrix Free?**
   Yes, the Voluntrix is absolutely free.

2. **What happens if the Voluntrix doesn't run?**
   Make sure that your JAVA plug-in is installed and functioning properly. If this doesn't work, use the command prompt to start the program (*Refer to the How to Startup section of the User's Manual).*

3. **Is there any way to add more than 30 projects to a Volunteer.**
   In the current version of the Voluntrix, there is space for only 30 projects for each volunteer. To handle this, it is wise to lump one or more projects together into one category and sum up their hours. *Note: The Voluntrix was developed for small, simple volunteering organizations; therefore, the program was designed to handle only a specific amount of data per volunteer.*

4. **How come there is no Address Field?**
   For the current version of the Voluntrix, there is no address field because the end users this program was specifically designed for did not see a need for this field; however, if the need for an address is necessary, the address of a volunteer can be placed instead of his/her funny fact.

D.3.20:Support

If you have questions, comments, or suggestions, do not hesitate to contact the developer, Prasanna Ranganathan, at prasannaranga@hotmail.com. You can expect your question/comment/suggestion to be handled within one business week via email.

D4. HTML Documentation

Found in downloadable compressed file